Week 4: Data Analysis

Algorithms (2)

Topics

- Linear regression
  - idea
  - model
  - RSS exact minimization
  - example
- Decision trees
- Logistic regression
  - idea
  - model
  - maximum likelihood training
  - example
- Neural networks
  - idea
  - model
  - RSS local minimization
  - example

Linear regression

Models

- Consider an input-output data set of N objects

$$D = \{(X_i, Y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^{N}$$

where;

$X_i \in \mathcal{X}$, $i = 1, \ldots, N$ are identical random variables (idem $Y_i \in \mathcal{Y}$).

- We assume there is a relationship between Y and X,

e.g. $\quad Y_i = f(X_i) + \varepsilon_i, \quad i = 1, \ldots, N$

where,

- f is a fixed but unknown function of X and
- ε_i are a random error terms.
- We aim to find the unknown model, f,
  - using the information in a given data set $D \sim \mathcal{D}$.

Learning machines

- We assume the model to be a specific instance of a given learning machine

$$F : \mathcal{X} \times \Lambda \to \mathcal{Y}.$$

- This week we consider 3 parametric learning machines

1. linear regression models
   where; $\to \mathcal{X} \in \mathbb{R}^d$ $\to \Lambda \in \mathbb{R}^{d+1}$ & $\mathcal{Y} \in \mathbb{R}$ (regression)

2. logistic regression models
   where; $\to \mathcal{X} \in \mathbb{R}^d$ $\to \mathcal{Y} \in \{0, 1\}$ (classification)
   $\to \Lambda \in \mathbb{R}^{d+1}$

3. neural network models
   where; $\to \mathcal{X} \in \mathbb{R}^d$ and $\to \mathcal{Y} \in \mathbb{R}$ or $\mathcal{Y} \in 1, \dots, K$
   $\to \Lambda \in \mathbb{R}^d$ $\to$ (regression or classification)

We obtain the corresponding models
i.e. $\hat{f}(x) = F(x, \hat{\lambda})$, $\hat{\lambda} \in \Lambda$,
- using labelled data sets $\mathcal{D} \sim D$ (supervised learning)

Linear regression

Let $\mathcal{X} = \mathcal{Y} = \mathbb{R}$ and $\Lambda \in \mathbb{R}$

- A linear regression learning machine is

$$F(x, \lambda) = \lambda_0 + \lambda_1 x, \quad \lambda \in \Lambda = \mathbb{R}^2$$

- Given $\hat{\lambda}$, predictions will be made by following the straight line

$$\hat{f}(x) = \hat{\lambda}_0 + \hat{\lambda}_1 x = \tilde{x}^T \hat{\lambda} = \langle \tilde{x}^T, \hat{\lambda} \rangle, \quad \tilde{x} = [1, x]^T$$

The dimensionality of Λ measures
- the flexibility of the model
  i.e. its degrees of freedom.

Least Squares training

- We train the model by minimizing the Empirical Risk, i.e. the RSS of the training set D

For $F: \mathbb{R} \times \mathbb{R}^2 \to \mathbb{R}$ we have

$$\hat{\lambda} = \arg \min_{\lambda \in \mathbb{R}^2} \sum_{(x,y) \in D} (F(x,\lambda) - y)^2$$

$$\hat{\lambda} = \arg \min_{\lambda \in \mathbb{R}^2} \sum_{(x,y) \in D} (\lambda_0 + \lambda_1 x - y)^2$$

- In this case, $\hat{\lambda}$ is usually called least squares parameter.

- The Empirical Risk
  - is an approximation of the (unavailable)
  - expected test error

$$E_D((\hat{f}(X) - Y)^2)$$

Example:

A synthetic data set generated by adding Gaussian noise, i.e. $\varepsilon \sim N(0, \sigma_\varepsilon)$, to a linear model

$$f_{true}(x) = \tilde{x}^T \lambda_{true}$$

A 2-dimensional parabola

For linear regression learning machines,
- the least squares objective

$$f(D, \lambda) = \sum_{(x,y) \in D} (\lambda_0 + \lambda_1 x - y)^2$$

has a nice shape and finding its global minimum (x) is easy.

Finding the global minimum

- If λ_0 is a minimizer of $f(D, \lambda)$ the gradient of f at λ_0 vanishes,

$$\nabla f(D, \lambda_0) = [0, 0]^T$$

- It is enough to compute $\nabla f(D, \lambda)$, for all $\lambda \in \Lambda$, and solve a linear system of equations,
  i.e. the vector equation above.

- The gradient of f at λ is

$$\nabla f(D, \lambda) = \sum_{(x,y) \in D} 2(\tilde{x}^T \lambda - y)\tilde{x} \quad , \quad \tilde{x} = [1, x]^T$$

A more compact notation

The objective function can be rewritten as

$$f(X, Y, \lambda) = \lambda^T X X^T \lambda + Y^T Y - 2\lambda^T X Y$$

where;

- $X \in \mathbb{R}^{(d+1) \times N}$ is now a $(d+1) \times N$ matrix &
- $Y \in \mathbb{R}^N$ a N-dimensional vector.

Rewriting ∇f

In this compact notation, the gradient of f is

$$\nabla f(X, Y, \lambda) = 2XX^T\lambda - 2XY$$

and the optimum is the solution of the vector equation.

$$XX^T\lambda = XY \quad , \quad \lambda = (XX^T)^{-1} XY$$

where;

$(XX^T)^{-1}$ is a matrix such that $(XX^T)^{-1}(XX^T) = I$

where,

I is the identity matrix.

Higher dimensions

- The matrix formulation applies to any linear regression model
  - with $X \in \mathbb{R}^d$, $\lambda \in \mathbb{R}^{d+1}$ and $d > 2$
- In $d = 2$, the vector equation can be written down in components and you obtain the system of equations

$$\hat{\lambda}_1 = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n} (x_i - \bar{x})^2} \qquad \hat{\lambda}_0 = \bar{y} - \hat{\lambda}_1 \bar{x}$$

Where;

- $\bar{x} = |D|^{-1} \sum_{(x,y) \in D} x \approx E(X)$ and
- $\bar{y} = |D|^{-1} \sum_{(x,y) \in D} y \approx E(Y)$

Example ①

Show that the formulas in the previous slides are equivalent to the vector equation $\nabla J = 0$

Hint: use $$[\nabla J]_0 = \sum_{(x,y) \in D} 2(\lambda_0 + x\lambda_1 - y)$$

$$[\nabla J]_1 = \sum_{(x,y) \in D} 2(\lambda_0 + x\lambda_1 - y)x$$

and

$$\hat{Var}(X) = |D|^{-1} \sum_{(x,y) \in D} (x - \bar{x})^2 = |D|^{-1} \sum_{(x,y) \in D} x^2 - \bar{x}^2$$

Example ②

Show that the same formulas can be obtained from the matrix equation $\lambda = (XX^T)^{-1} XY$ using

$$XX^T = \sum_{(x,y)\in D} \begin{pmatrix} \tilde{x}_0^2 & \tilde{x}_0\tilde{x}_1 \\ \tilde{x}_0\tilde{x}_1 & \tilde{x}_1^2 \end{pmatrix} \qquad XY = \sum_{(x,y)\in D} \begin{pmatrix} \tilde{x}_0 y \\ \tilde{x}_1 y \end{pmatrix}$$

and the definition of the inverse matrix.

Hint: for 2x2 matrices one has inverse, i.e. A^{-1}

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

Extension: polynomial regression

- Linear regression models
  - can be extended for capturing possible polynomial dependencies.
- The attribute vector $x \in \mathcal{X}$
  - is extended to include higher powers of its entries, e.g.

$$x = [x_0, x_1, \ldots, x_d] \to x_{ext} = [x_0, x_1, \ldots, x_d, x_1^2, \ldots, x_d^2]$$

- The parameter space is adapted accordingly, e.g. $\Lambda \to \Lambda_{ext} \subseteq \mathbb{R}^{2d+1}$ and the learning process is an in the linear case.

Extension: nominal attributes

- Linear regression models
  - can be extended for accepting at the same time continuous and nominal attributes, e.g. $\mathcal{X} = C \times \mathbb{R}$ with $C = \{blue, red, yellow\}$

- $x_1 \in C$ is a true nominal variable. i.e. there is no underlying ordering, and it does not make sense to replace C with $\{1, 2, 3\}$ or similar.

- The idea is to replace the nominal variable, $x_1 \in C$, with $|C| - 1$ boolean variables, e.g. $x_1 \to [x_b, x_r]$

where;

$$x_b = 1[x = yellow], \quad x_r = 1[x = red]$$

- As the number of numerical attributes is now $|\mathcal{X}| = d + |C| - 2$ the parameter space should also expanded.
- For $\mathcal{X} = C \times \mathbb{R}$ with $C = \{blue, red, yellow\}$ the learning machine becomes

$$F(x, \lambda) = \lambda_0 + \lambda_b x_b + \lambda_r x_r + \lambda_2 x_2$$

As a result, we train base model,

$$f_{yellow}(X_2) = \lambda_0 + \lambda_2 X_2,$$

Which is valid only for the yellow objects and two shifted models,

$$f_{red}(X_2) = f_{yellow} + \lambda_r \;\&\; f_{blue}(X_2) = f_{yellow} + \lambda_b$$

which,

- are valid for the remaining red & blue objects.